Web Application Projects versus Web Site Projects in Visual Studio

.NET Framework 4.5

In Visual Studio you can create *web application projects* or *web site projects*. You create or open a web application project by choosing **New Project** or **Open Project** in the Visual Studio **File** menu. You create or open a web site project by choosing **New Web Site** or **Open Web Site** in the File menu. It's best to choose the right type before you create a web project, because it can be time-consuming, difficult, and error-prone to convert from one type to the other.

Note

For new development, we recommend that you choose web application projects. This topic explains that web site projects have some advantages, but many developers who choose web site projects eventually find that the disadvantages outweigh any perceived advantages. In addition, as new ASP.NET features are developed, they won't always be made available for web site projects. For example, the next Visual Studio release after Visual Studio 2012 will have new tooling for creating web projects, and this new tooling will work only with web application projects. For more information, see Creating an ASP.NET Web Project in Visual Studio 2013.

This topic contains the following sections:

Scenarios

Scenarios in which web application projects are the preferred choice include the following:

- You want to be able to use the Edit and Continue feature of the Visual Studio debugger.
- You want to run unit tests on code that is in the class files that are associated with ASP.NET pages.
- You want to refer to the classes that are associated with pages and user controls from standalone classes.
- You want to establish project dependencies between multiple web projects.
- You want the compiler to create a single assembly for the entire site.
- You want control over the assembly name and version number that is generated for the site.
- You want to use MSBuild or Team Build to compile the project. For example, you might want to add prebuild and postbuild steps.
- You want to avoid putting source code on a production server.

Scenarios in which Web site projects are the preferred choice include the following:

- You want to include both C# and Visual Basic code in a single web project. (By default, a web application is compiled based on language settings in the project file. Exceptions can be made, but it is relatively difficult.)
- You want to open the production site in Visual Studio and update it in real time by using FTP.
- You do not want to have to explicitly compile the project in order to deploy it.
- If you do precompile the site, you want the compiler to create multiple assemblies for the site, which can include one assembly per page or user control, or one or more assemblies per folder.
- You want to be able to update individual files in production by just copying new versions to the production server, or by editing the files directly on the production server.
- If you precompile the site, you want to be able to update individual ASP.NET web pages (.aspx files) without having to recompile the entire site.
- You like to keep your source code on the production server because it can serve as an additional backup copy.

Summary of Differences

The following table summarizes the main differences.

Area	Web application projects	Web site projects
Project file structure	A Visual Studio project file (.csproj or .vbproj) stores information about the project, such as the list of files that are included in the project, and any project-to-project references.	There is no project file (.csproj or .vbproj). All the files in a folder structure are automatically included in the site.
Compilation	• You explicitly compile the source code on the computer that is used for development or source control. • By default, compilation of code files (excluding .aspx and .ascx files) produces a single assembly.	• The source code is typically compiled dynamically (automatically) by ASP.NET on the server the first time a request is received after the site has been installed or updated. You can precompile the site (compile in advance on a development computer or on the server). • By default, compilation produces multiple assemblies.
Namespaces	Explicit namespaces are added to pages, controls, and classes by default.	Explicit namespaces are not added to pages, controls, and classes by default, but you can add them manually.
Deployment	• You copy the assembly to a server. The assembly is produced by compiling the application. • Visual Studio provides tools that integrate with Web Deploy (the IIS web deployment tool) to automate many deployment tasks.	• You copy the application source files to a computer that has IIS installed on it. • If you precompile the site on a development computer, you copy the assemblies produced by compilation to the IIS server. • Visual Studio provides tools that integrate with Web Deploy (the IIS web deployment tool) to automate many deployment tasks.

Project File Structure

Web application projects use Visual Studio project files (.csproj or .vbproj) to keep track of information about the project. This makes it possible to specify which files are included in or excluded from the project, and therefore which files are compiled during a build.

For web site projects, all files in a folder structure are automatically considered to be included in the web site. If you want to exclude something from compilation, you must remove the file from the web site project folder or change its file-name extension to an extension that is not compiled and is not served by IIS.

An advantage of using project files in web application projects is the following:

- It is easy to temporarily remove files from the site but still make sure that you do not lose track of them, because they remain in the folder structure. For example, if a page is not ready to be deployed, you can temporarily exclude it from the build without deleting it from the folder structure. You can deploy the compiled assembly, and then include the file in the project again. This is especially important if you are working with a source control repository.

An advantage of using folder structure without project files in Web site projects is the following:

- You do not have to manage the project's structure exclusively in Visual Studio. For example, you can copy files into the project or delete them from the project by using File Explorer.

Compilation

For web application projects, you typically build the project in Visual Studio or by using the ASP.NET batch compiler on a computer that is not the production IIS server. All code-behind class files and standalone class files in the project are compiled into a single assembly, which is then put in the web application project's Bin folder. (The .aspx and .ascx files are compiled dynamically in a manner similar to what is done for web site projects.)

For web site projects, you do not have to manually compile the project. web site projects are typically compiled dynamically by ASP.NET (on both the development computer and the production IIS server). You can choose between batch compilation mode, which typically produces one assembly per folder, and fixed compilation mode, which typically produces one assembly for each page or user control.

Advantages of the compilation model for web application projects include the following:

- You can use MSBuild to create a custom batch-compilation process.
- It is easy to specify assembly attributes such as name and version.
- Compiling in advance makes sure that users do not have to wait while the site compiles on the production server. (If the site is very large, dynamic compilation of a web site project might take a noticeable amount of time. Dynamic compilation occurs when a request for a site resource is received after an update to the site, and the request that triggers compilation might be delayed while the required resources are compiled. If the delay is unacceptable, you can precompile the site. However, then some of the advantages of dynamic compilation are lost.)
- You have complete control over where you put code files in the project folder structure, and how you how classes in the project refer to each other. (Dynamic compilation requires that the source code for any classes that are used throughout the site must be in the App_Code folder. You cannot refer to a page or user control class from a class in App_Code.)

Advantages of the compilation model for Web site projects include the following:

- You can test specific pages regardless of the state of other pages. This is because running an individual page does not require that the whole site compile successfully, only the page and any components it depends on, such as code in the App_Code folder or the Global.asax file. (In a web application project, if there are compilation errors anywhere in the site, you cannot create the assembly and therefore cannot test even the pieces of the site that compile.)

- It is easy to update a Web site in production. You can update individual source code files on the production server without having to explicitly recompile the site. You can update individual files that are ready for deployment even if other files are not ready due to compile errors. You can also open the Web site on the production IIS server directly in Visual Studio and update the Web site in real time.

- Precompiling to multiple assemblies can have a performance advantage in some scenarios. A typical example is a site that has many pages with lots of code written for them. Most of the pages are rarely requested and only some are used frequently. If you compile a site like this into multiple assemblies, the production server can load only the assemblies that are required for the current requests. If a page is not requested, its corresponding assembly is not loaded.

Note

There is no difference in performance between a web site project and a web application project. The only significant exceptions are the ones that have already been noted, and as a practical matter they apply only to very large sites. The first request to the web site might require the site to be compiled, which can result in a delay. And if the web site is running on an IIS server that is short on memory, including the entire site in a single assembly might use more memory than would be required for multiple assemblies.

Deployment

You typically copy all of your source code to the web server when you deploy a web site project. In a web application project, source code is compiled into an assembly (a .dll file), and that is what has to be on the web server.

An advantage of web application projects is that you can avoid deploying source code to the IIS server. In some scenarios, such as shared hosting environments, you might be concerned about unauthorized access to source code on the IIS server. For a web site project, you can avoid this risk by precompiling on a development computer and deploying the generated assemblies instead of the source code. However, in that case you lose some of the benefits of easy site updates.

An advantage of web site projects is that when you make a small change to a web site, you do not have to redeploy the whole site. Instead, you can copy just the changed file or files to the production IIS server. You can also edit files directly on the production server. Because a web application project's code files are compiled into a single assembly file, you must deploy the whole site even for small changes, unless the only change is to an .aspx or .ascx file.

Deployment often involves other tasks in addition to copying assemblies or code to a server. For example, database scripts might have to run in production, and connection strings in the Web.config file might have to be changed for a production server. Visual Studio provides tools such as one-click publish that work with web application projects to automate many of these tasks. These tools are available for both web application projects and web site projects. For more information, see Web Deployment Overview for Visual Studio and ASP.NET.

See Also

Concepts

Web Deployment Overview for Visual Studio and ASP.NET

Other Resources

Web Deployment Content Map for Visual Studio and ASP.NET

Web Application Projects vs Web Site Projects

© 2016 Microsoft